

Cloud Computing

Virtual Resources and Distributed Cloud Applications

Chapter 8: Beyond Cloud



Distributed and Operating Systems
Electrical Engineering and Computer Science
Technische Universität Berlin

Overview

8.1 Federated, Hybrid and Multi-Clouds

8.2 Fog and Edge

8.3 Cloudlets

8.4 Approximate Computing

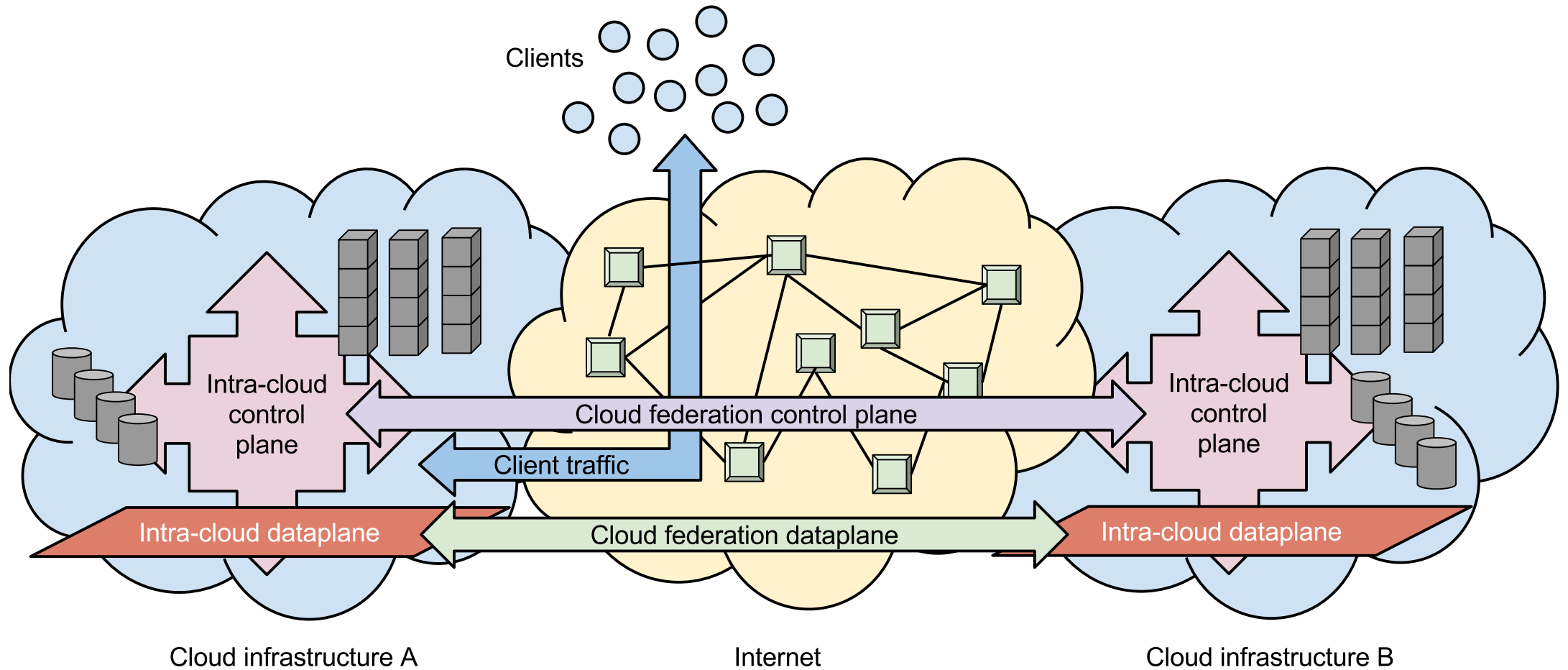
8.1 Extended Cloud Concepts

- Combining providers and models
- Hybrid Cloud
 - Mixing public and private cloud models
- Multi Cloud
 - Using multiple cloud providers in one application
- Federated Cloud
 - Integrating multiple cloud providers to provide a service

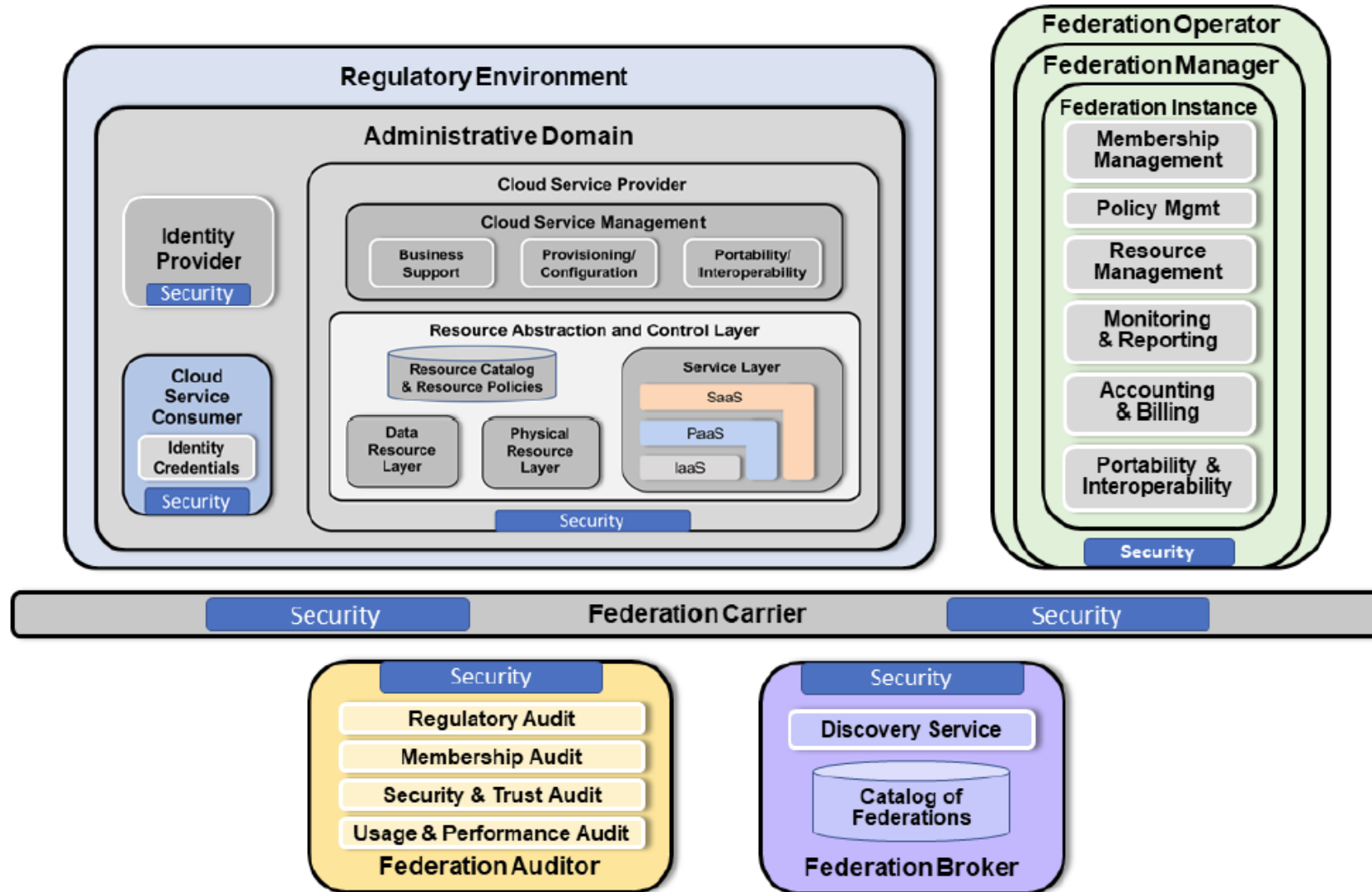
Motivation for Cloud Federation

- Resilience against vendor outages
- Fighting vendor lock-in
- Harness the diversity of cloud services by different providers
- Harness different locations of providers
- Compliance with policies
 - i.e., data needs to be stored in EU
 - Needs a way to express and enforce the policies

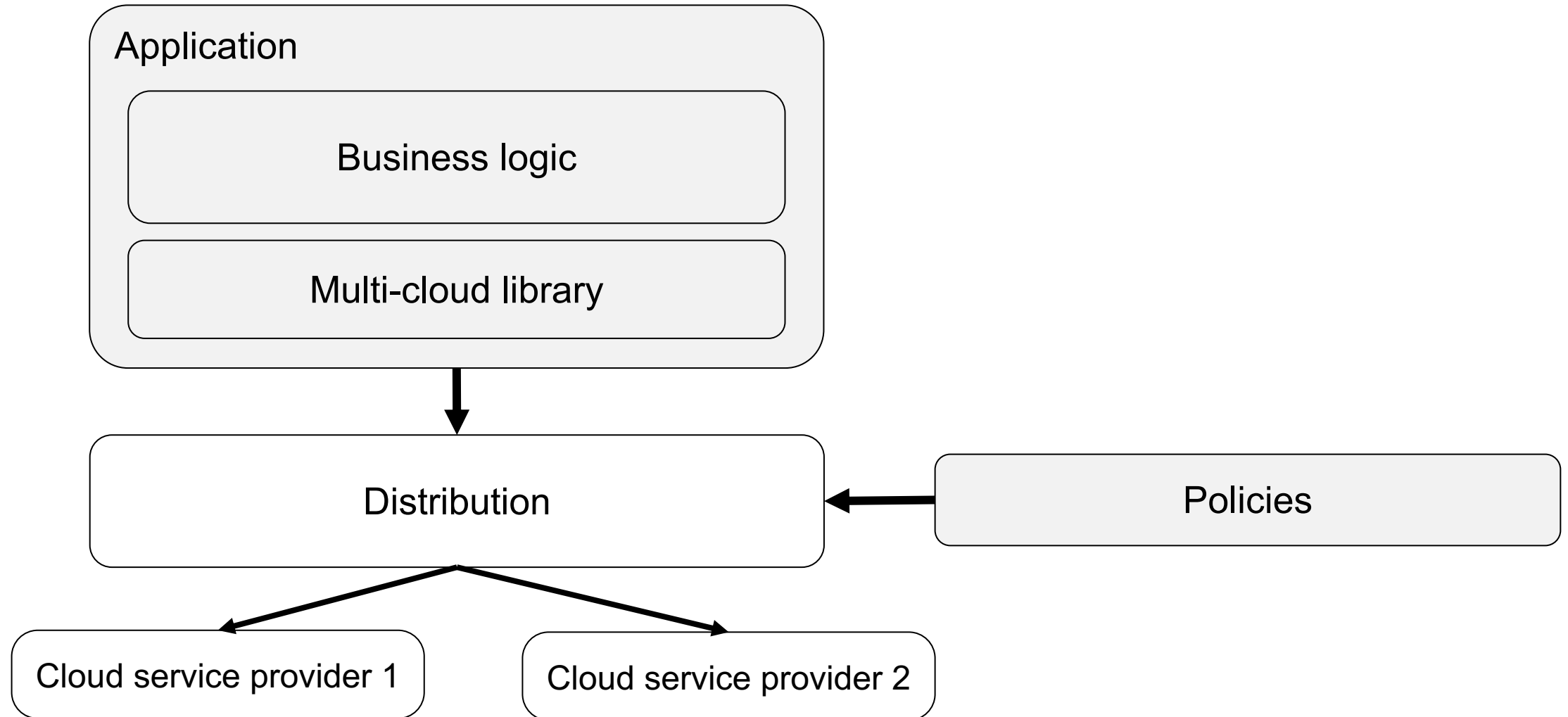
Cloud Federation



NIST Cloud Federation Reference Architecture



Cloud Federation



Policy Enforcement

- Paper from 2018

Giving Customers Control over Their Data: Integrating a Policy Language into the Cloud

Jens Hiller*, Maël Kimmerlin†, Max Plauth‡, Seppo Heikkilä§,

Stefan Klauck‡, Ville Lindfors¶, Felix Eberhardt‡, Dariusz Bursztynowski||,

Jesus Llorente Santos†, Oliver Hohlfeld*, Klaus Wehrle*

*Communication and Distributed Systems, RWTH Aachen University, Germany

†School of Electrical Engineering, Aalto University, Finland

‡Hasso Plattner Institute for Digital Engineering, University of Potsdam, Germany

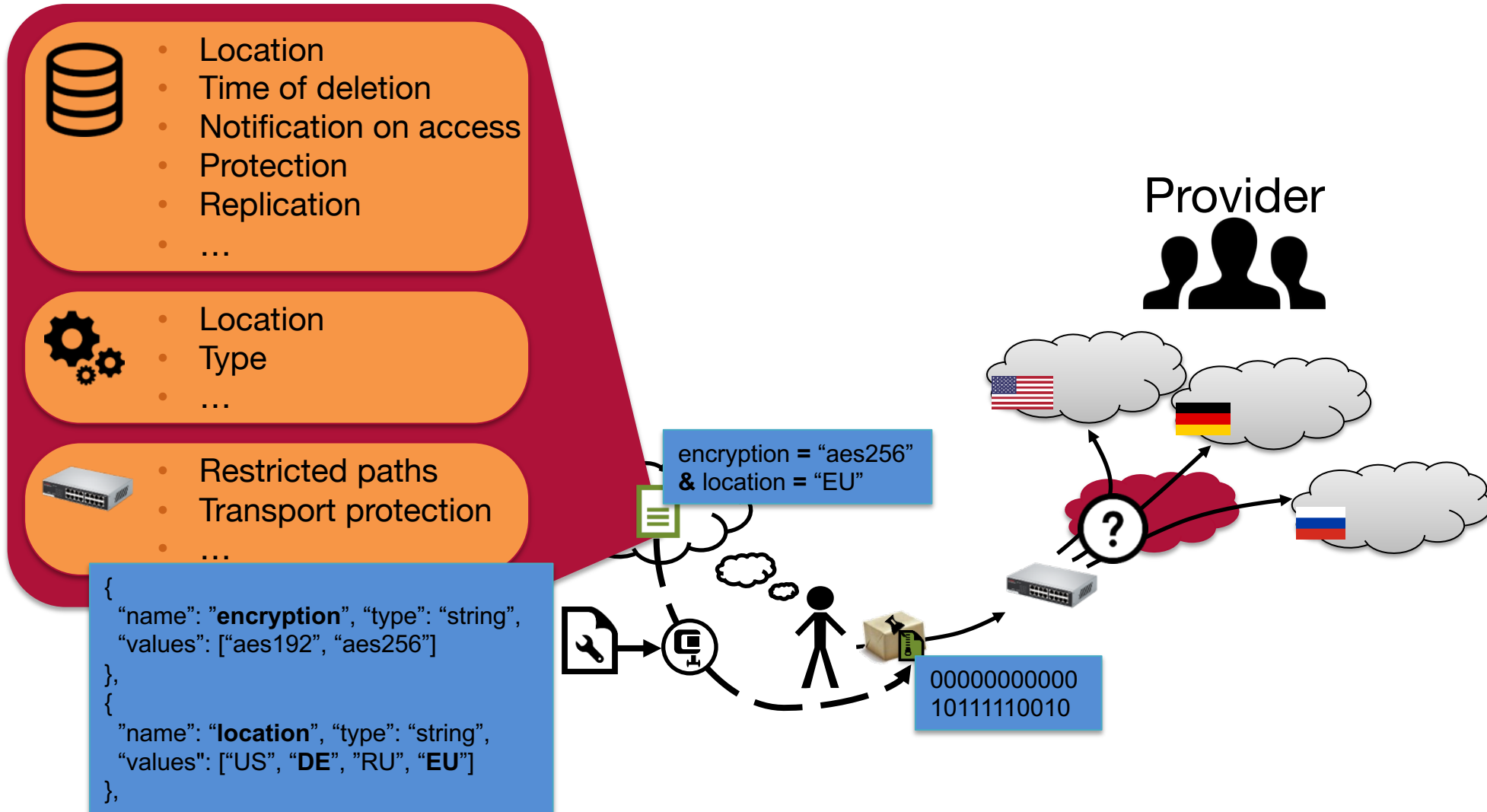
§Helsinki Institute of Physics, CERN, Geneva, Switzerland

¶F-Secure Oyj, Finland ||Orange Polska S.A., Poland **equal contribution**

- Idea: Showing how the CPPL policy language could be used for all service models of the Cloud (IaaS, PaaS, SaaS)

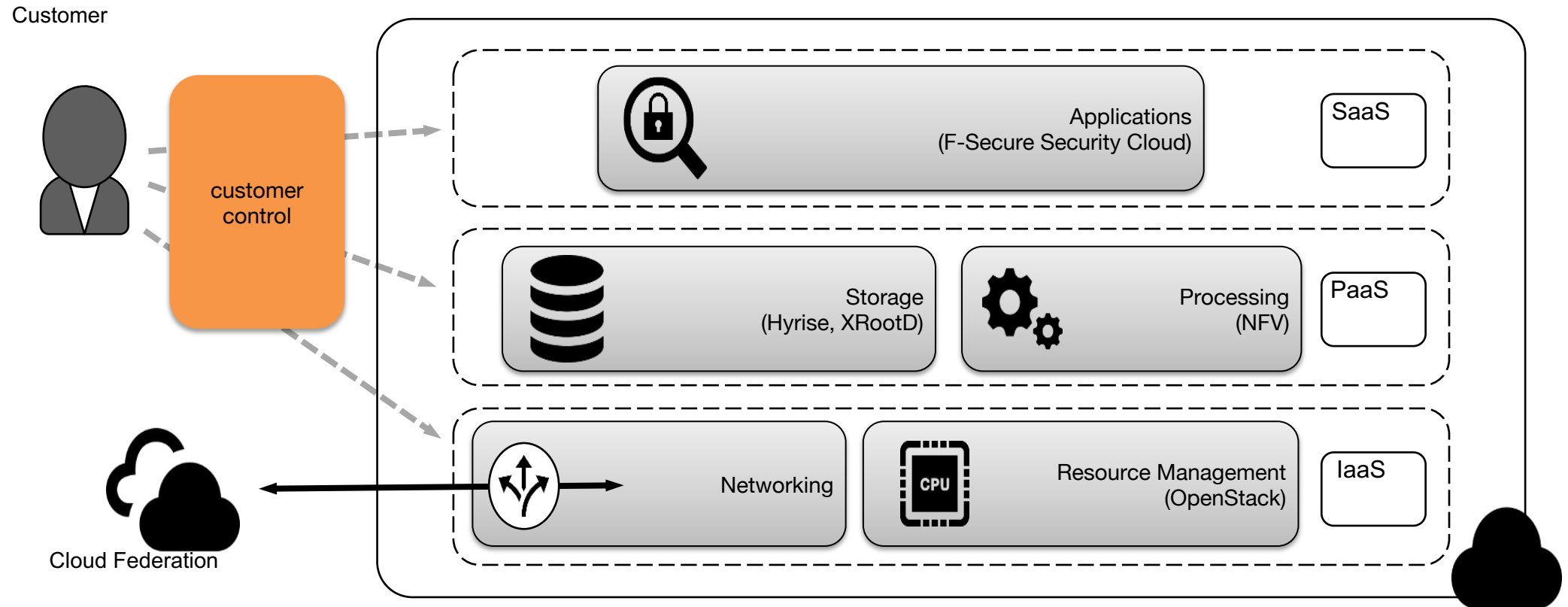
Customer Requirements

CPPL Policy Language



Policy-Aware Cloud Architecture

- Policy-awareness must be addressed at all cloud layers



OpenStack – Policy-Aware Resource Management

The screenshot displays the OpenStack Policy-Aware Resource Management web interface. The top header includes the OpenStack logo, a 'demo' dropdown menu, and a user profile 'demo' with a dropdown arrow. The left sidebar contains navigation links: 'Project', 'Identity', 'Settings' (highlighted), 'User Settings', 'Change Password', 'Edit Policy' (highlighted with a red bar), and 'Developer'. The main content area is titled 'Edit Policy' and features a 'Policy' section with a JSON configuration in a text editor. The configuration includes 'availability_zones' with a value of 'az2' and 'storage' with 'encryption' set to 'true'. To the right of the editor is a 'Description:' field with the text 'Edit your policy.' At the bottom right of the main area is a blue 'Save' button. A user menu is open on the right side, showing options for 'Settings', 'Help', 'Themes' (with 'Default' selected and 'Material' as an option), and 'Sign Out'.

openstack demo demo

Project Identity Settings User Settings Change Password Edit Policy Developer

Edit Policy

Edit Policy

Policy

```
{
  "availability_zones": [
    "az2"
  ],
  "storage": {
    "encryption": true
  }
}
```

Description:
Edit your policy.

Save

Settings Help Themes: Default Material Sign Out

OpenStack – Policy-Aware Resource Management

The screenshot shows the OpenStack Volumes dashboard. The top header includes the OpenStack logo, a 'demo' dropdown menu, and a user profile 'demo' with a dropdown arrow. The left sidebar contains a 'Project' section and a 'Compute' section with links for 'Overview', 'Instances', 'Volumes' (highlighted in red), and 'Images'. The main content area is titled 'Volumes' and has three tabs: 'Volumes' (active), 'Volume Snapshots', and 'Volume Consistency Groups'. A red error message box in the top right corner states: 'Error: Your policy requires using an encrypted volume type.' Below the tabs, there is a search filter box, a '+ Create Volume' button, and an '⇌ Accept Transfer' button. A table with columns: Name, Description, Size, Status, Type, Attached To, Availability Zone, Bootable, Encrypted, and Actions is shown. The table body contains the text 'No items to display.'

openstack demo demo

Volumes

Volumes Volume Snapshots Volume Consistency Groups

Filter + Create Volume ⇌ Accept Transfer

Name	Description	Size	Status	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
No items to display.									

Error: Your policy requires using an encrypted volume type.

Overview

8.1 Federated, Hybrid and Multi-Clouds

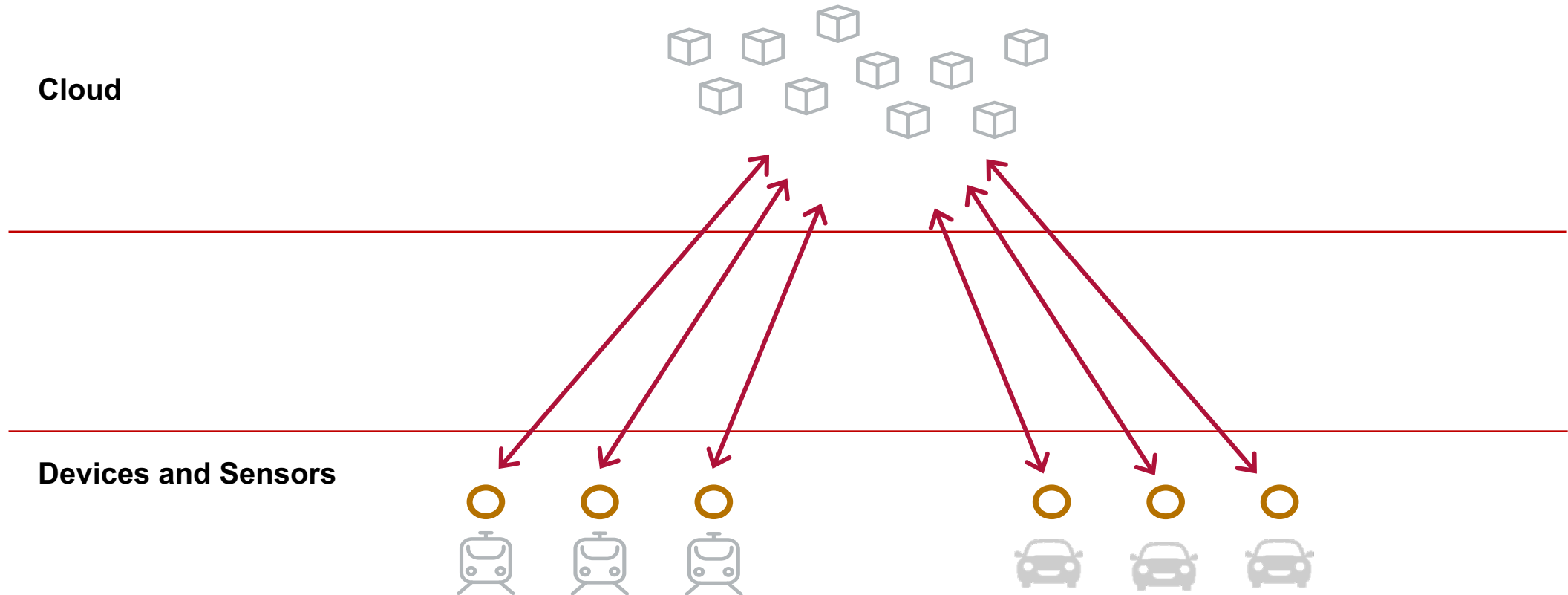
8.2 Fog and Edge

8.3 Cloudlets

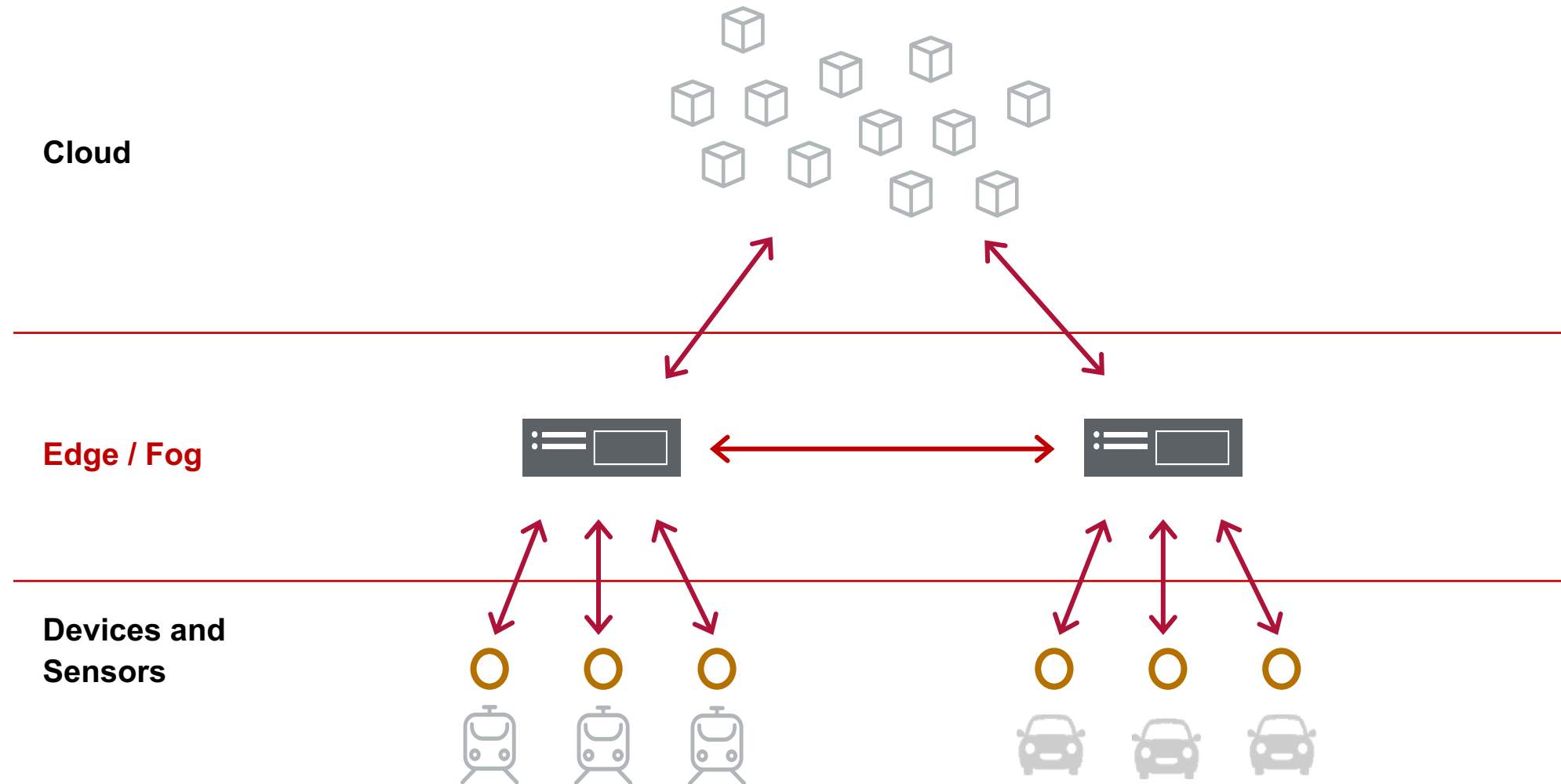
8.4 Approximate Computing

8.2 Cloud and the IoT

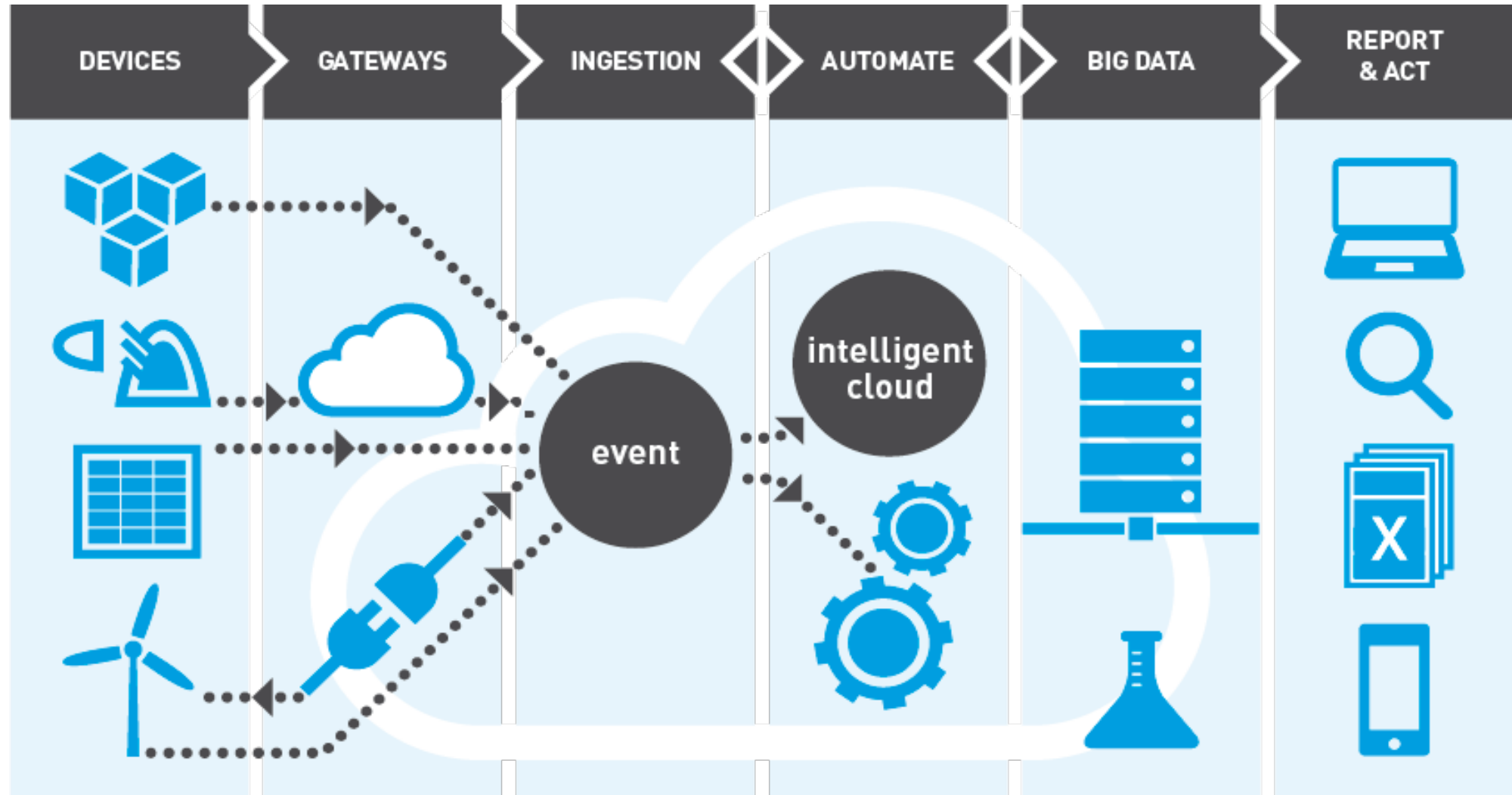
- Assumption: By 2023, 5.6 billion smart sensors and other IoT devices employed around the world generating >507.5 zettabytes of data.



Cloud and the IoT: Fog Computing

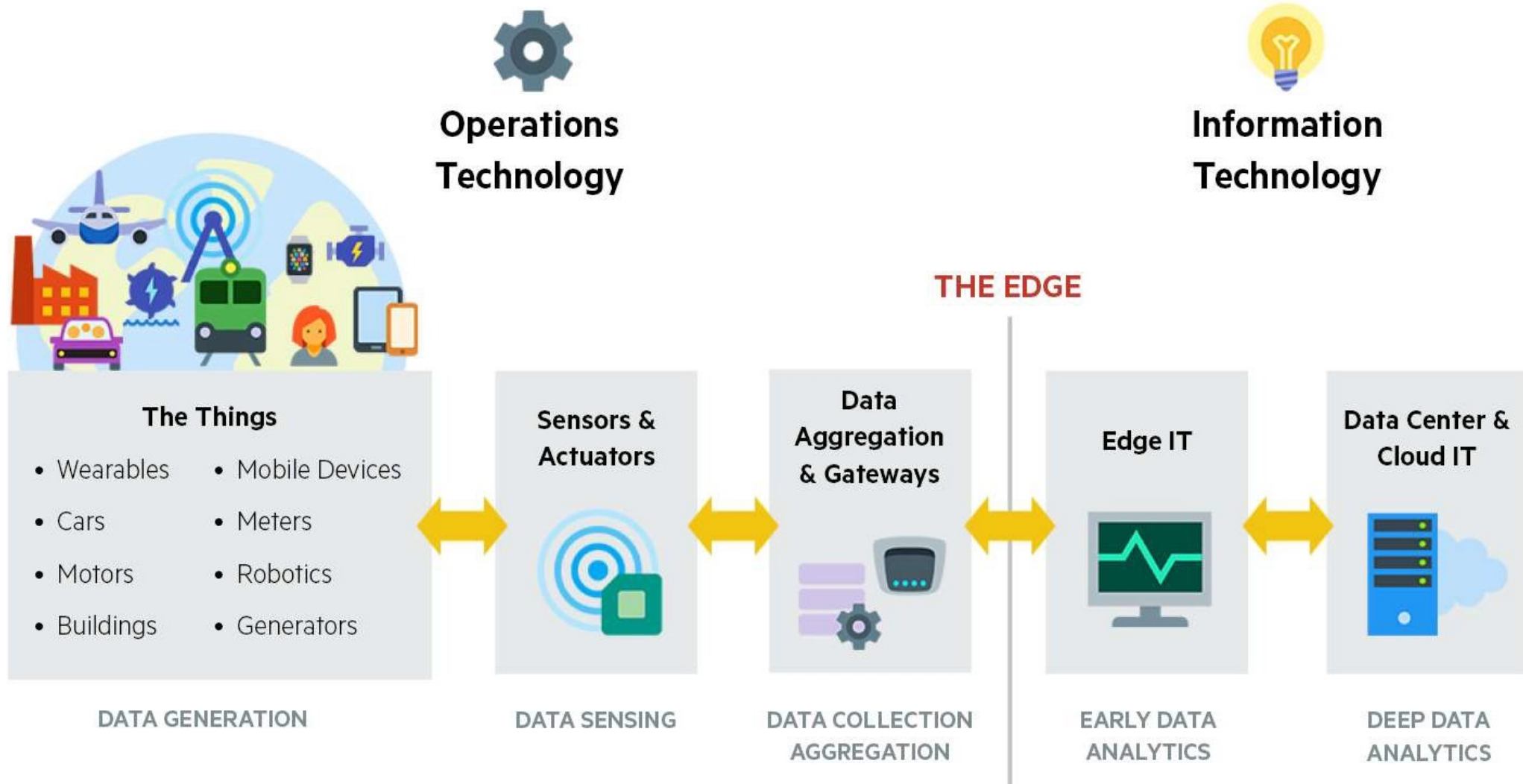


IoT End-to-End Value Chain

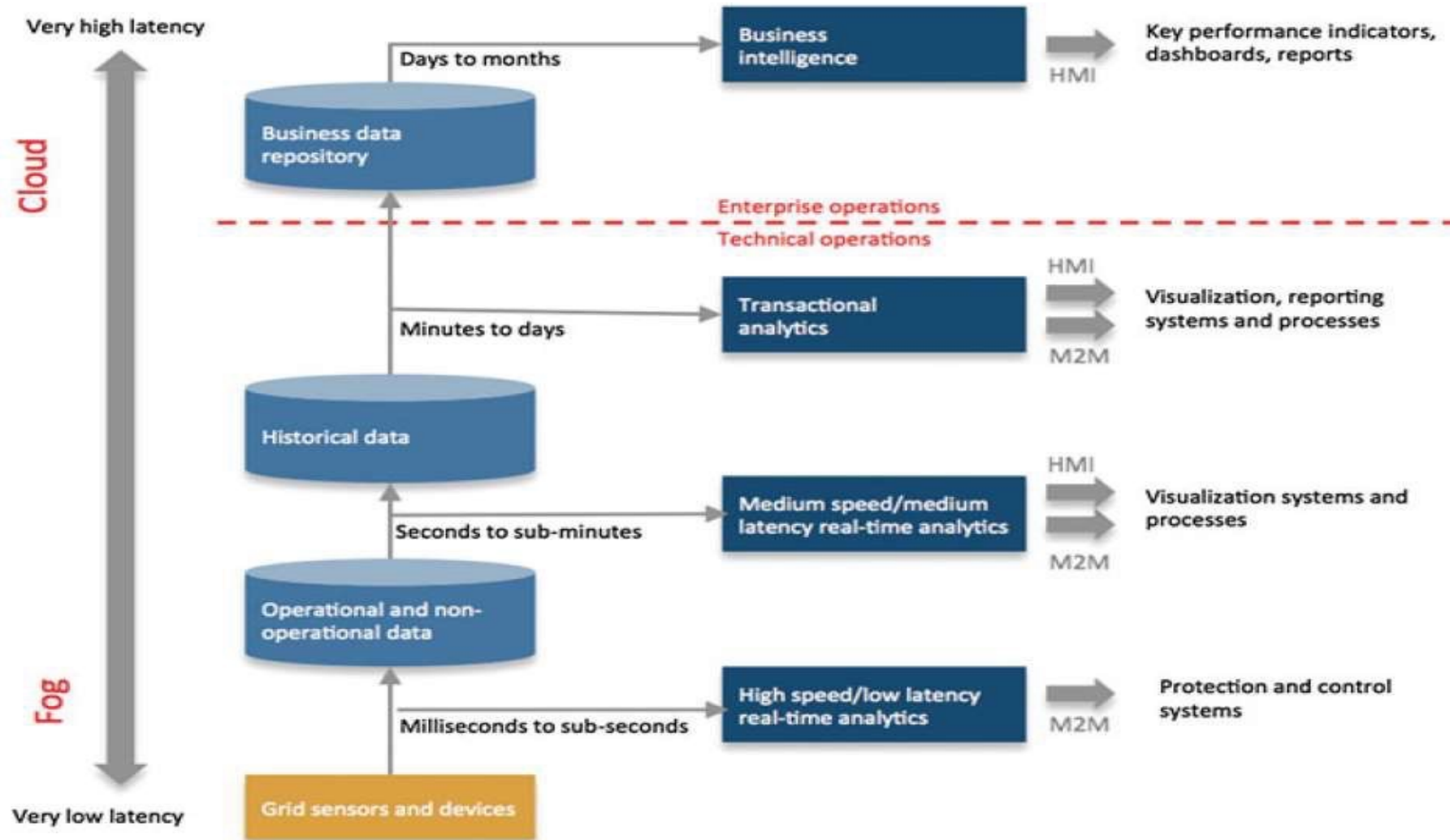


<https://www.cs.sjtu.edu.cn/~wuct/bdit/slides/lec4.pdf>

Fog/Edge Computing for Real Time Data



Different Requirements on Latency



Motivation for Fog Computing

- Challenges obstructing IoT devices' service quality include computational energy, battery durability, storage capacity, and bandwidth → Tight cooperation with cloud providers
- This model is prone to the inefficiency of basic computing and communication requirements
 - Location awareness
 - Mobility support / Geo-distribution
 - Low latency
- Motivation for Fog Computing according to OpenFog Consortium
 - IoT data is growing exponentially → network congestion and performance problems at the edge of infrastructure
 - Cloud-only solution is insufficient for many tasks due to factors such as performance, security, bandwidth, reliability→ Performance control (latency and efficiency) as primary concern in fog computing

NIST Definition of Fog Computing

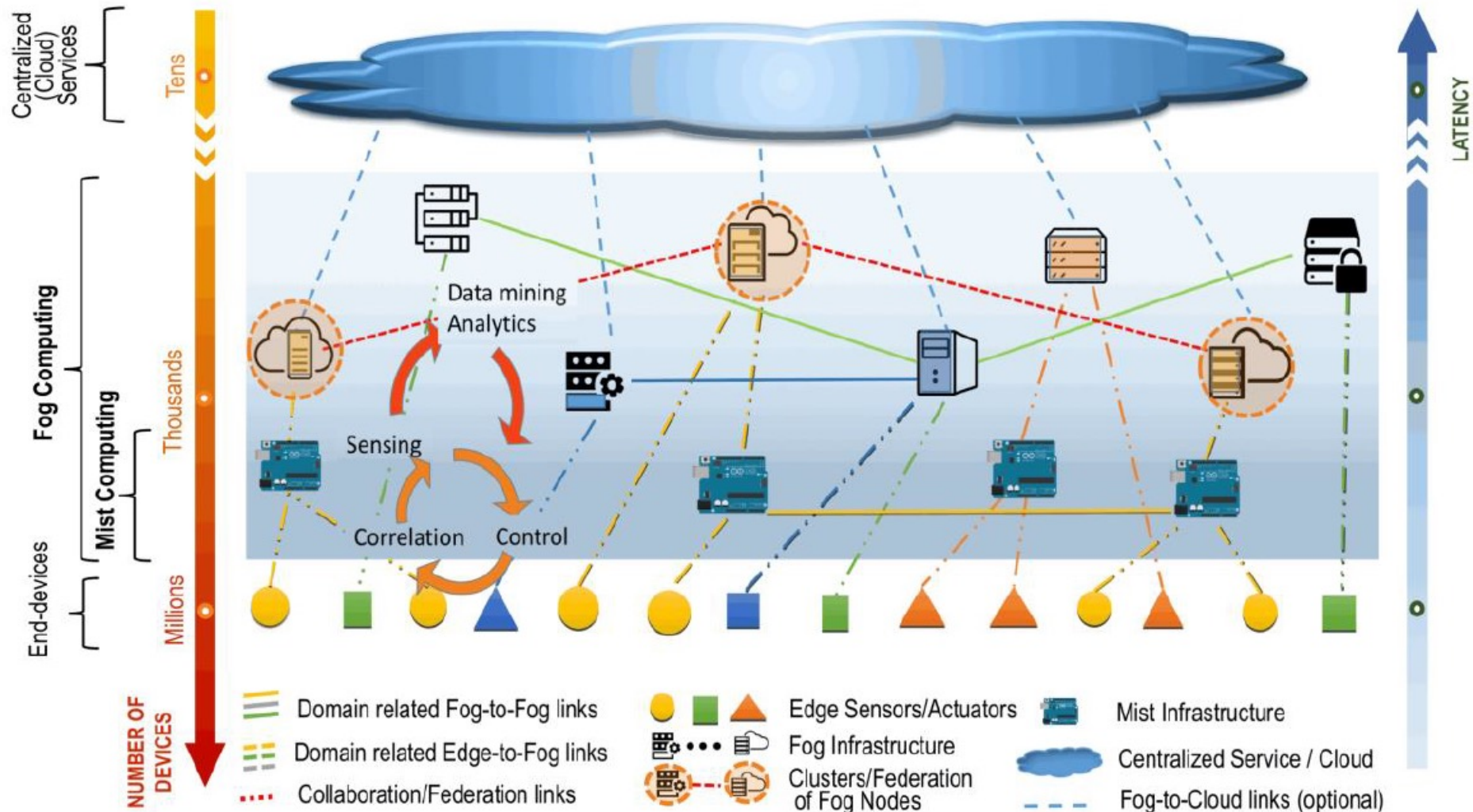
Fog computing is a layered model for enabling **ubiquitous access to a shared continuum of scalable computing resources**.

The model facilitates the **deployment of distributed, latency-aware applications and services**, and consists of **fog nodes (physical or virtual)**, residing **between smart end-devices and centralized (cloud) services**. The fog nodes are context aware and support a **common data management and communication system**.

Fog computing **minimizes the request-response time** from/to supported applications, and provides, for the end-devices, **local computing resources** and, when needed, **network connectivity to centralized services**.

Standard 500-325

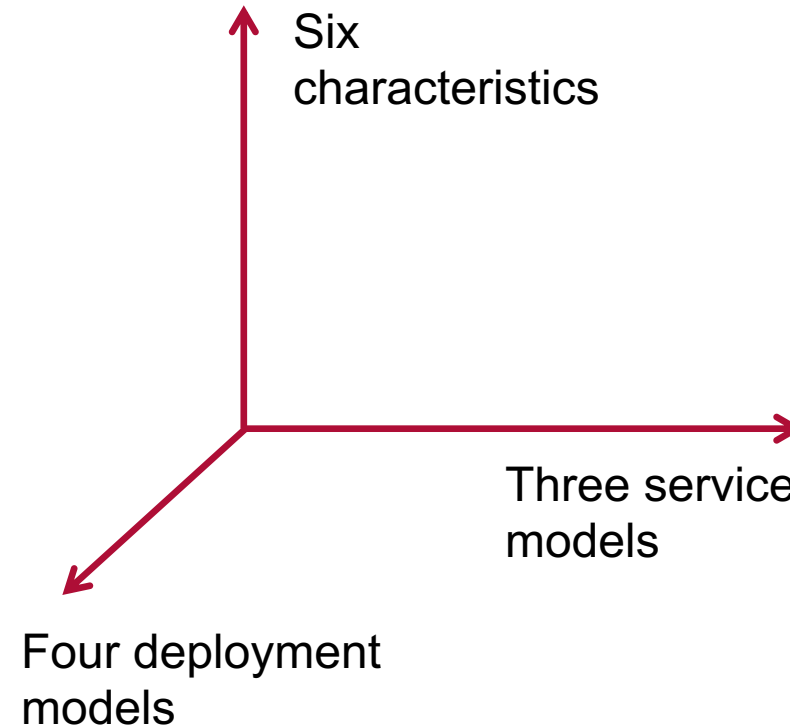
NIST Definition of Fog Computing (Ctnd.)



<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.500-325.pdf>

Dimensions of Fog Computing (NIST)

- Architectural service types:
 - SaaS
 - PaaS
 - IaaS
- Node deployment models:
 - private nodes
 - community nodes
 - public nodes
 - hybrid nodes



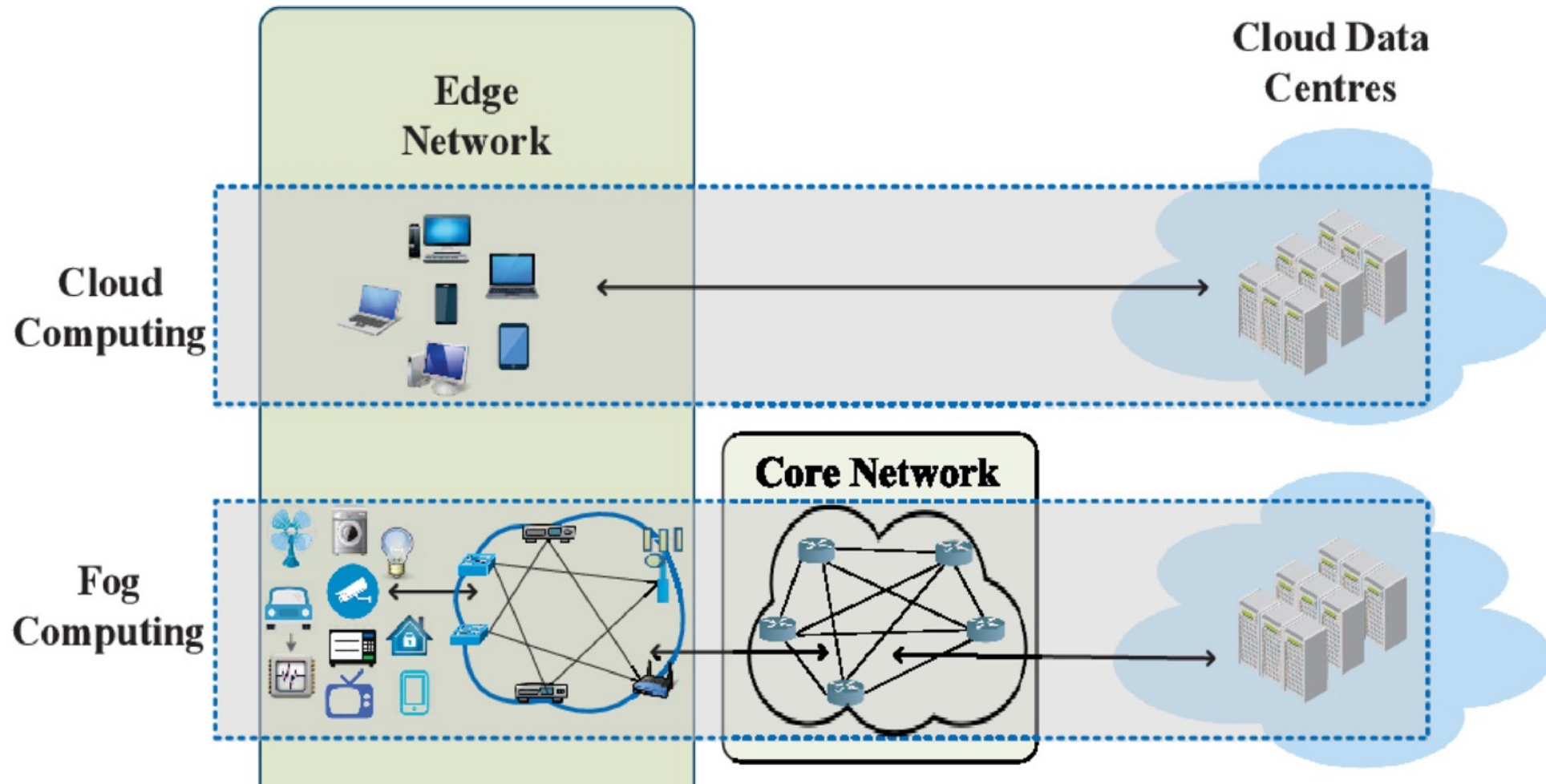
Characteristics of Fog Computing

- Contextual location awareness
- Low latency
- Geographical distribution
- Heterogeneity
- Interoperability and federation
- Real-time interactions
- Scalability and agility of federated, fog-node clusters

Resp. time	Subjective impression
<150ms	Crisp
150ms–1s	Noticeable to Annoying
1s–2s	Annoying
2s–5s	Unacceptable
> 5s	Unusable

Perception of response times

Computation domain of cloud and fog computing



Three Fog Layers

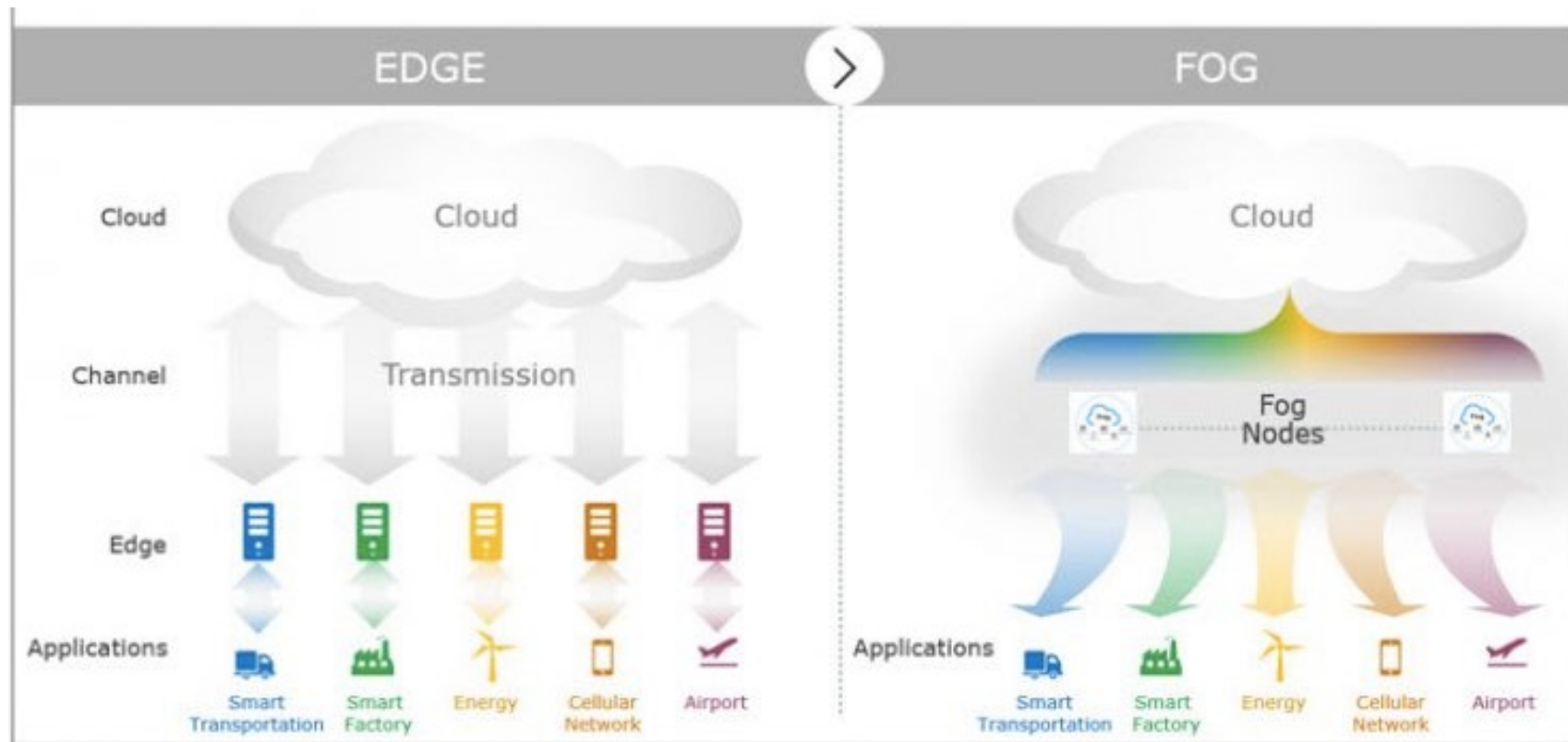
1. Terminal/Edge: geographically distributed end devices fetching data and conveying it to a higher-level layer for processing and storage.
Examples: wearables, sensors, intelligent vehicles, smartphones
2. Fog: contains the "fog nodes," e.g. a mobile or non-mobile node, placed in a fixed, strategic location.
Examples: access points, routers, servers, switches, base stations, etc. The computational ability optimizes the services of latency-sensitive applications, allowing real-time processing
3. Cloud: storage devices and servers with high performance and computational power responsible for executing non-latency-sensitive jobs sent by the lower fog layer.
Examples: DigitalOcean, iCloud, Google App Engine, Amazon EC2

Fog Offloading

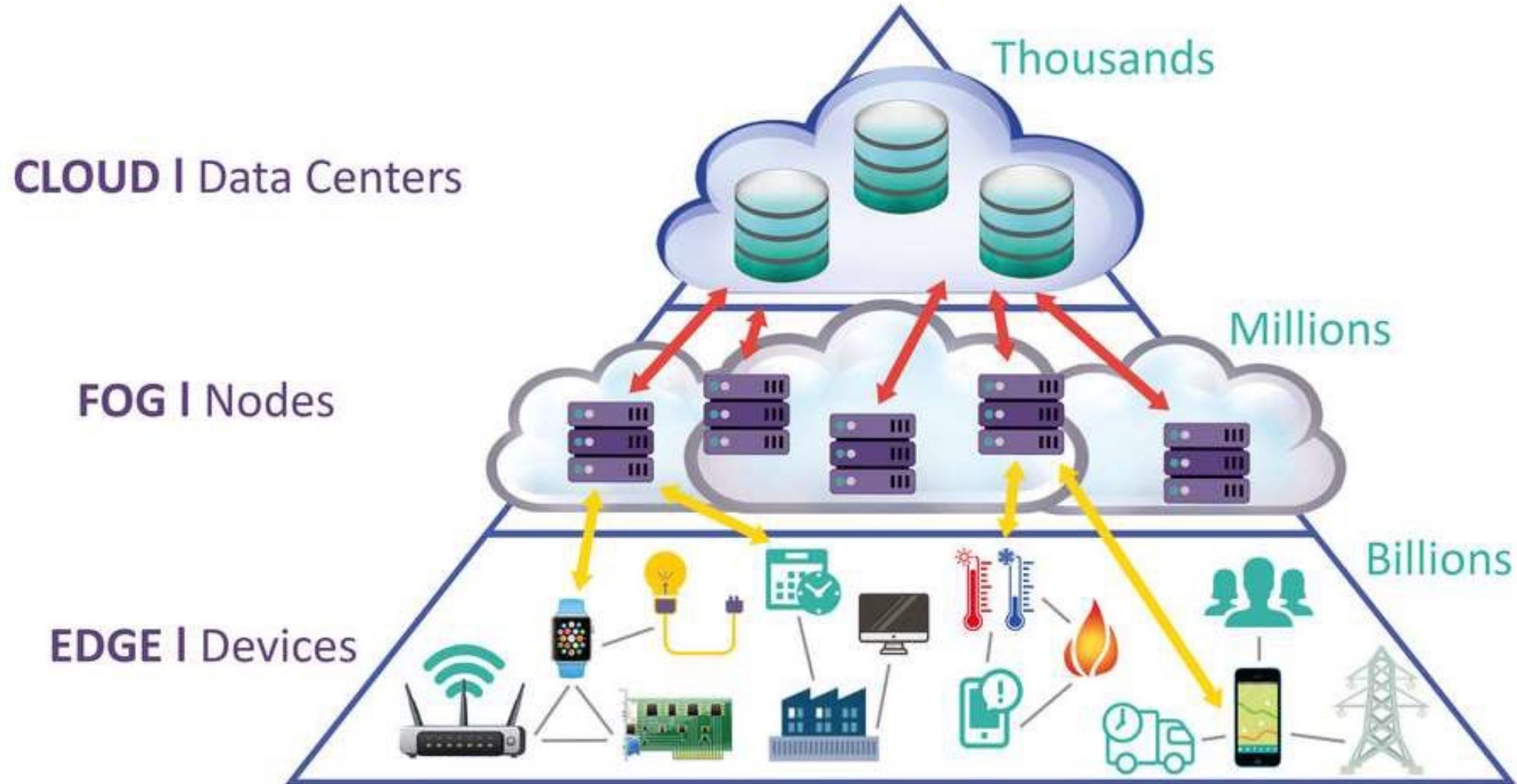
- Offloading = The process of moving data and tasks from devices (such as smartphones or IoT devices) to fog nodes, which are placed closer to the edge of the network
 - This makes processing quicker and less demanding on the cloud, improves the network's effectiveness and performance as a whole
 - increased security and privacy by keeping sensitive data inside the fog network
- Research and sophisticated algorithms for application partitioning between cloud and mobile devices, offloading decision, and distributed task execution

From Edge to Fog

- Fog computing complements edge computing by providing a layer of computing infrastructure between the edge devices and the cloud.

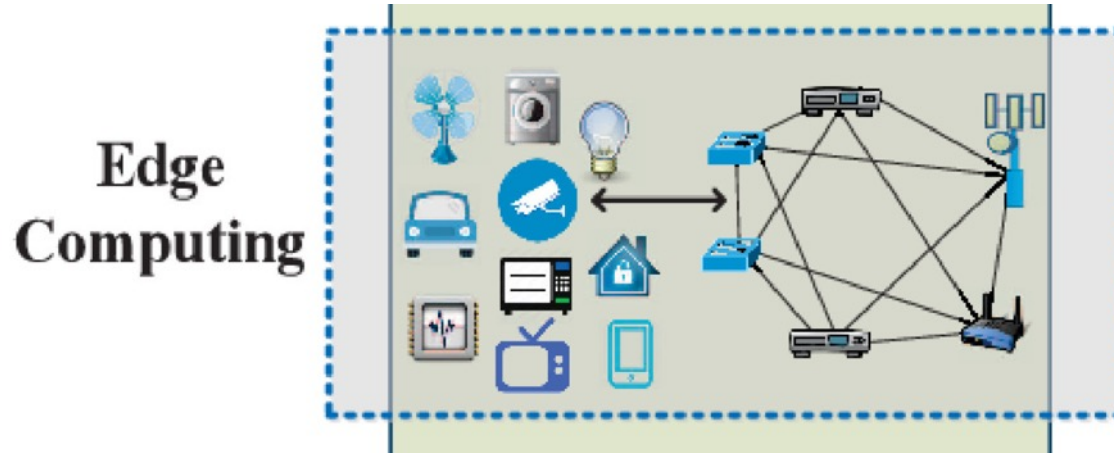


Cloud-Fog-Edge Architecture

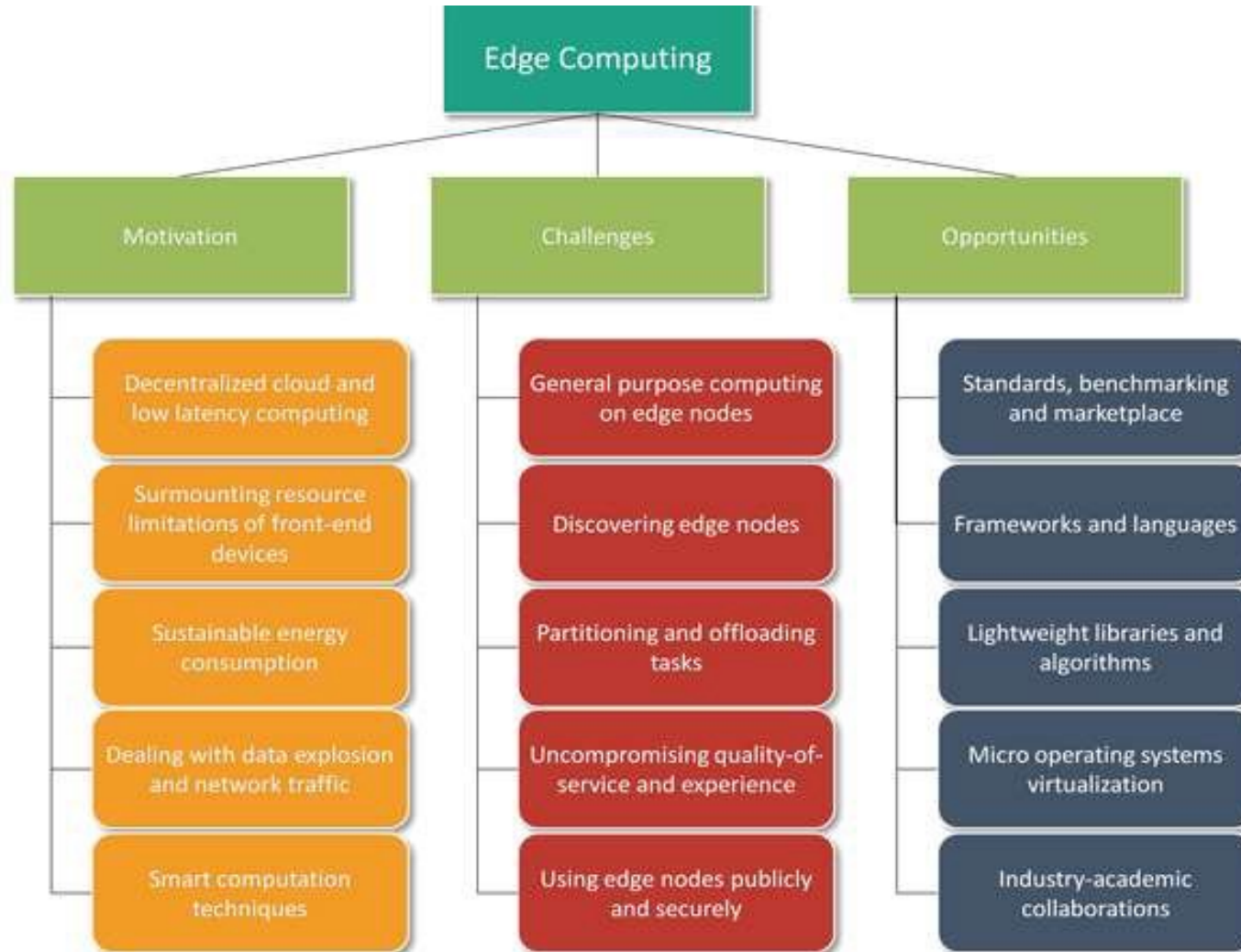


Related technologies: Edge Computing

- Computing outside the cloud on the edges of the network providing computation and data storage closer to the sources
 - improve response times, save bandwidth
 - architecture rather than a specific technology
- Origins in CDNs (content distribution networks, late 1990s) to serve web and video content from edge servers close to users

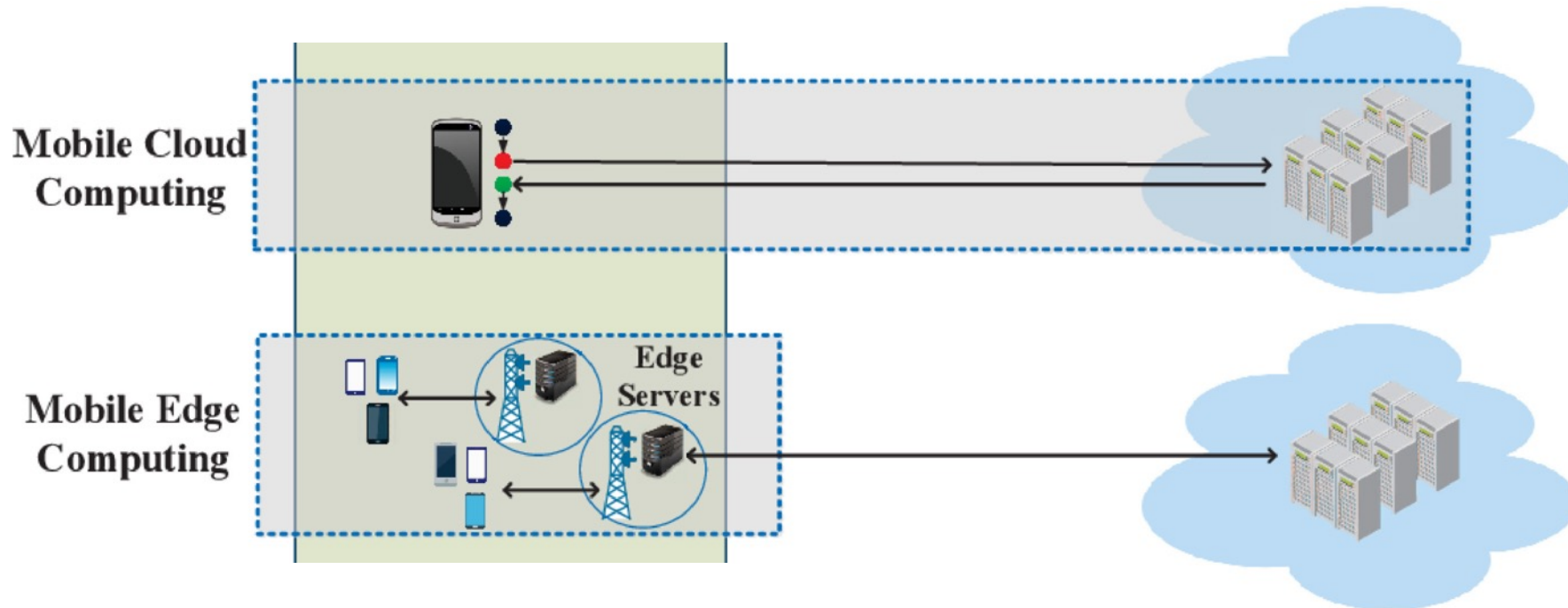


Challenges in Edge Computing

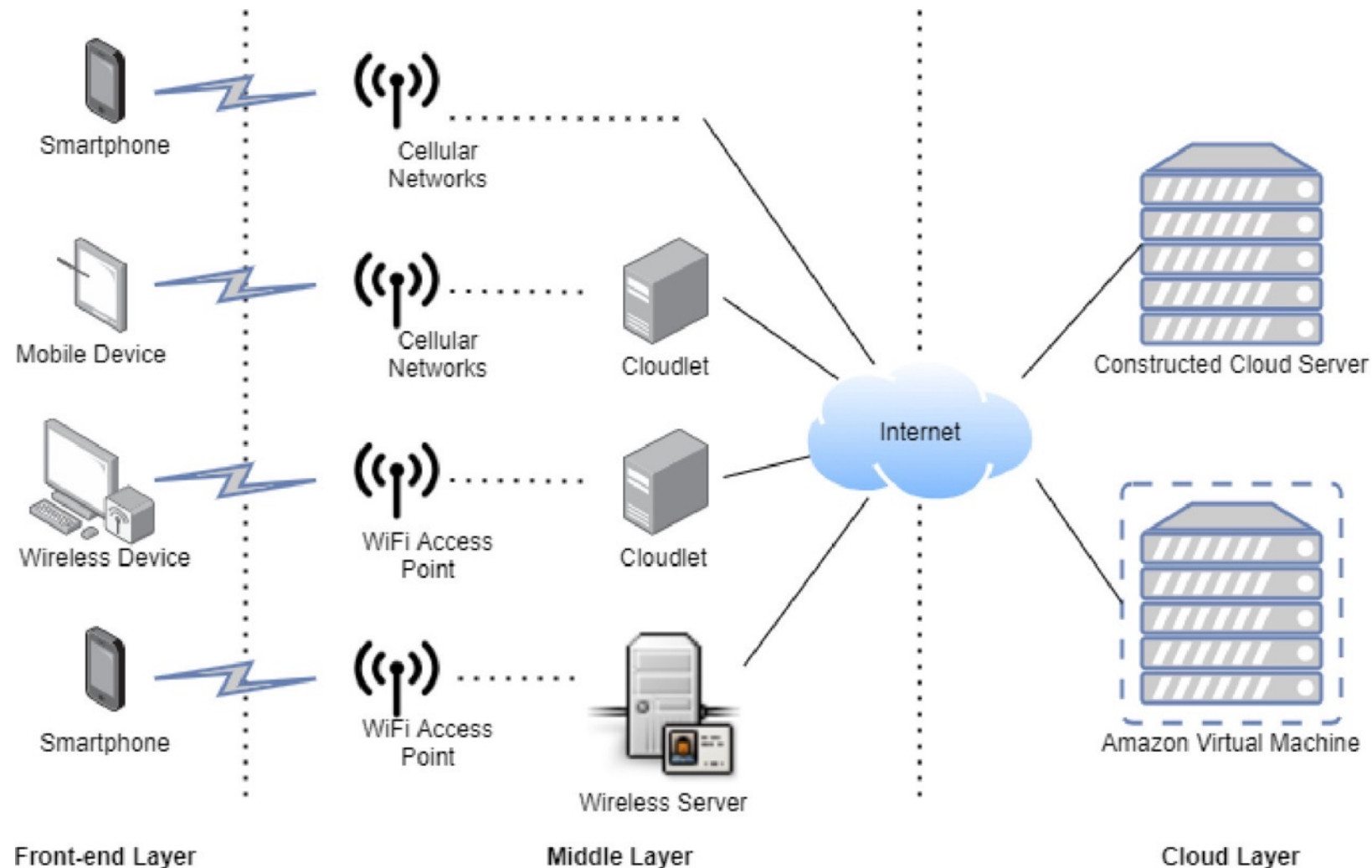


Related technologies: MEC and MCC

- Mobile Edge Computing (MEC): enabler of the current development of cellular base stations
- Mobile Cloud Computing (MCC) offers processing resources required to facilitate the remote execution of offloaded mobile applications closer to end users.



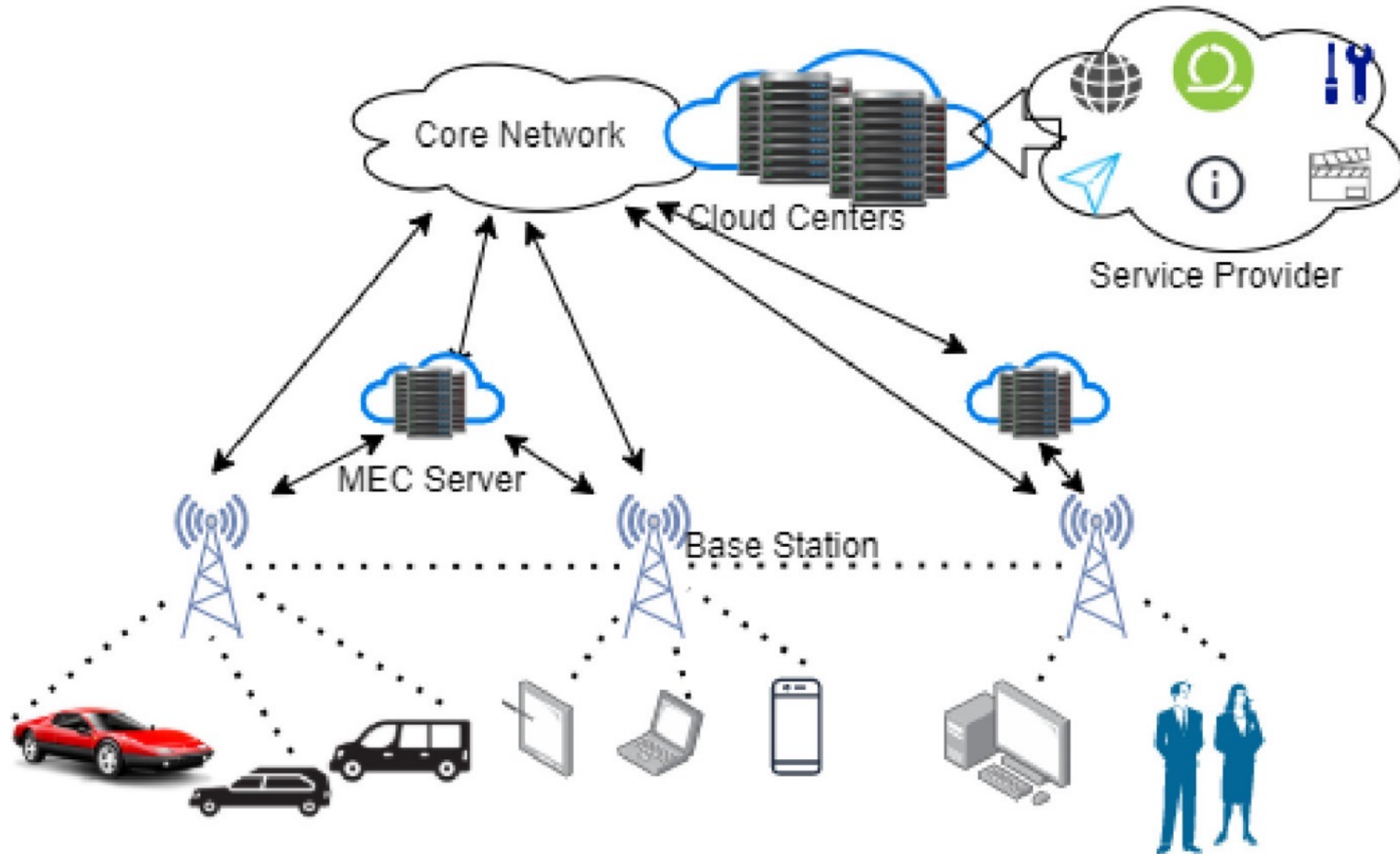
Mobile cloud computing architecture



Mobile cloud computing (MCC)

- MCC combines cloud computing, mobile internet, and mobile computing
 - Use of resources (network, server, mobile application, storage, and computing) based on a request in the mobile environment
 - In MCC architecture, the cloud servers are placed far away from the edge devices, making it inefficient in a network environment with high computational requirements
- Technology proposed to provide a new framework with services to mobile subscribers by taking maximum advantage of cloud computing
- MCC frequently deploys at the network's edge small, lightweight cloud servers known as cloudlets.

Mobile-edge computing architecture



Overview

8.1 Federated, Hybrid and Multi-Clouds

8.2 Fog and Edge

8.3 Cloudlets

8.4 Approximate Computing

8.3 Cloudlets

- Paper from 2009

The Case for VM-based Cloudlets in Mobile Computing

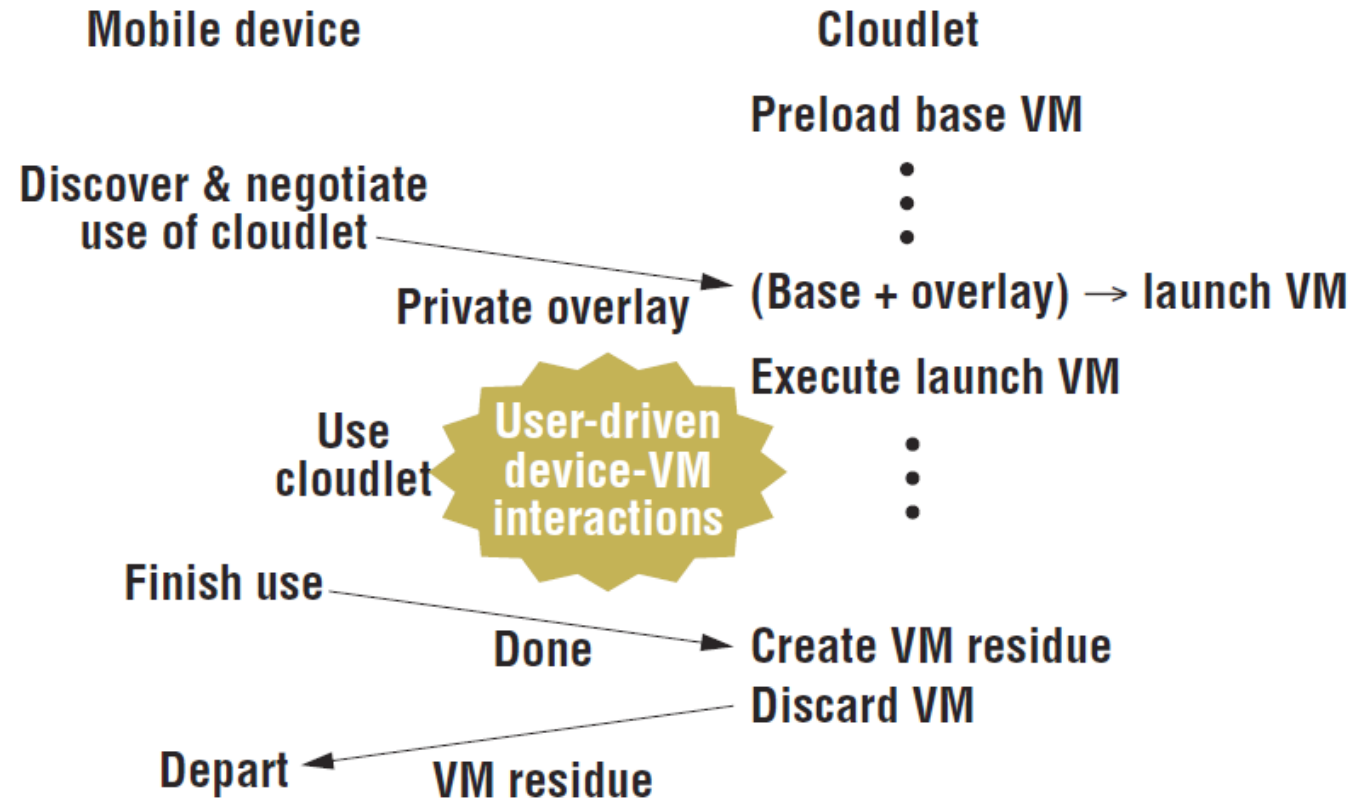
Mahadev Satyanarayanan[†], Paramvir Bahl[‡], Ramon Caceres[•], Nigel Davies[°]

[†]Carnegie Mellon University, [‡]Microsoft Research, [•]AT&T Research, [°]Lancaster University

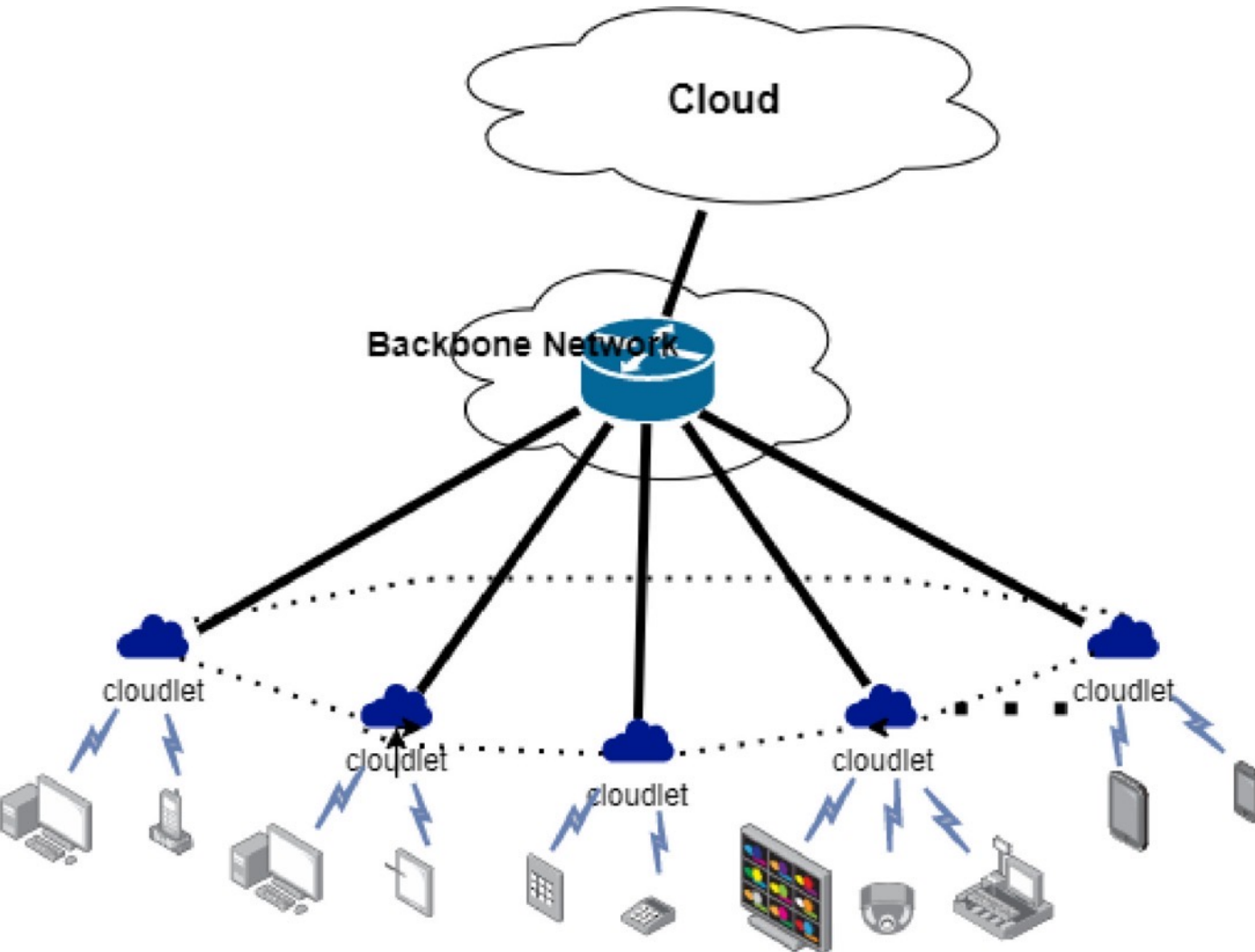
- Idea: deploy micro data centers (called cloudlets) everywhere for usage in mobile applications
- Cloudlet = tiny box data center, mounted at a wireless hop's distance from the mobile devices in a public place such as hospitals, office buildings, and shopping malls, to ease an appropriate application
 - Cluster of powerful computers with high-speed internet access and a high-bandwidth wireless LAN for use by surrounding mobile devices
 - Housed in a tamper-resistant box for safety reasons

Concept

- Concept 1: VM migration
 - Concept 2: Dynamic VM synthesis
 - Mobile device transmits the VM overlay to the cloudlet, which applies it to the base VM to generate the launch VM
 - Assumption: small number of base VMs (few releases of Linux and Windows configurations) will be popular worldwide in cloudlets
- odds high that a mobile device will find a compatible base for its overlays



Cloudlet Architecture



	Cloudlet	Cloud
State	Only soft state	Hard and soft state
Management	Self-managed; little to no professional attention	Professionally administered, 24X 7 operator
Environment	“Datacenter in a box” at business premises	Machine room with power conditioning and cooling
Ownership	Decentralized ownership by local business	Centralized ownership
Network	LAN latency/ bandwidth	Internet latency/ bandwidth
Sharing	Few users at a time	100s-1000s of users at a time

Overview

8.1 Federated, Hybrid and Multi-Clouds

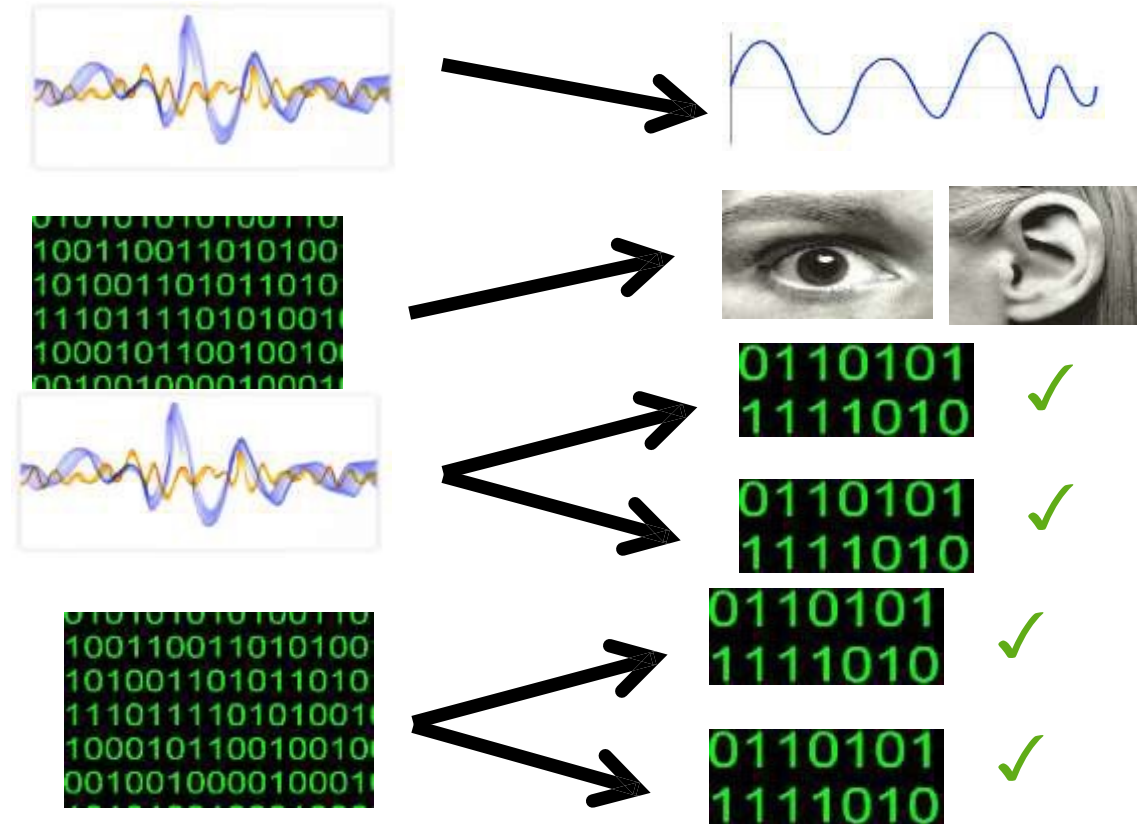
8.2 Fog and Edge

8.3 Cloudlets

8.4 Approximate Computing

8.4 Approximate Computing

- Image, sound and video processing
- Simulations, games, search, machine learning
- Image rendering
- Sensor data analysis, computer vision
- Inexact/imprecise input data
- Approximate/iterative algorithms
- Loose constraints on output



Where a lot of (most?) resources go!

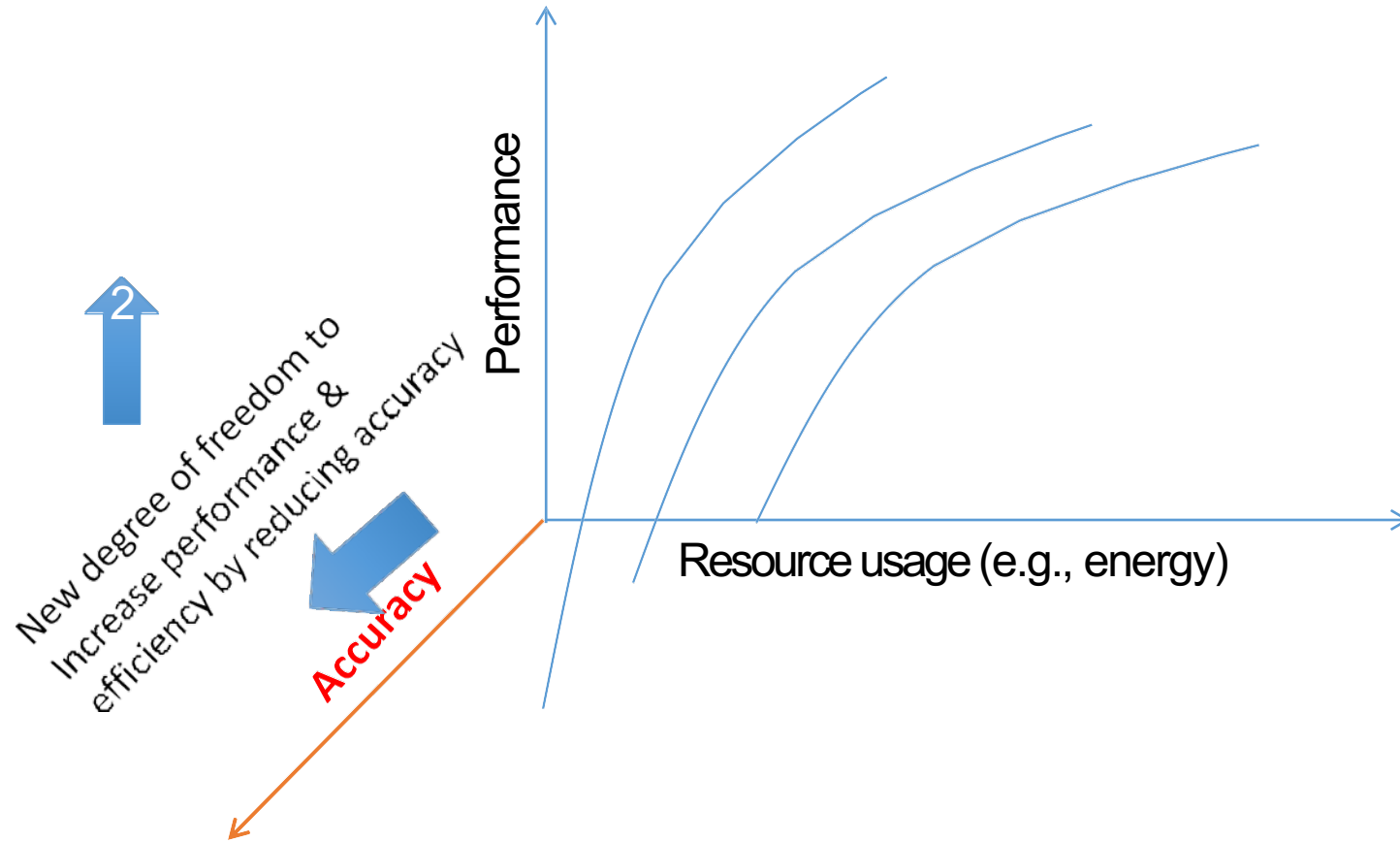
Approximate computing

- Emerging paradigm for energy-efficient and/or high-performance design
 - A possibly inaccurate result sufficient for its purpose rather than a guaranteed accurate result in order to save time / energy
 - Observation: bounded approximation can provide disproportionate gains in performance and energy, while still achieving acceptable result accuracy
- Application in non-critical, error-tolerant domains (e.g. multimedia, machine learning, signal processing, scientific computing and other applications related to human perception/cognition and have inherent error resilience)
- Example
 - Search engine where no exact answer may exist for a certain search query and hence, many answers may be acceptable.
 - Occasional dropping of some frames in a video application can go undetected due to perceptual limitations of humans.
 - K-means clustering: 5% loss in classification accuracy can provide 50 times energy saving compared to the fully accurate classification.

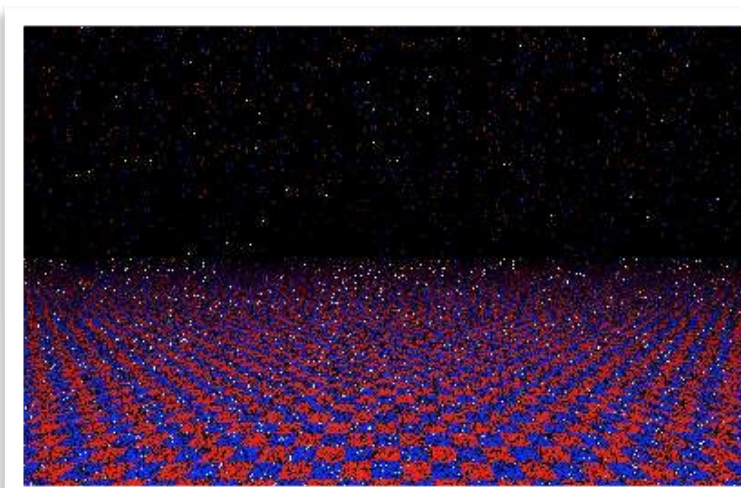
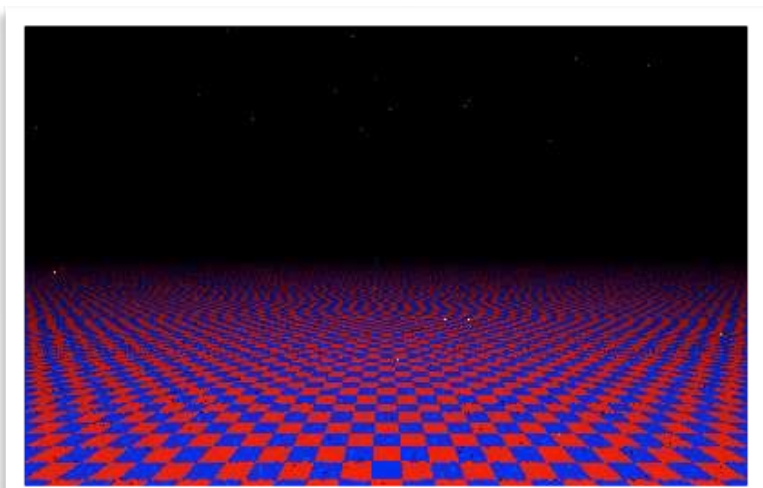
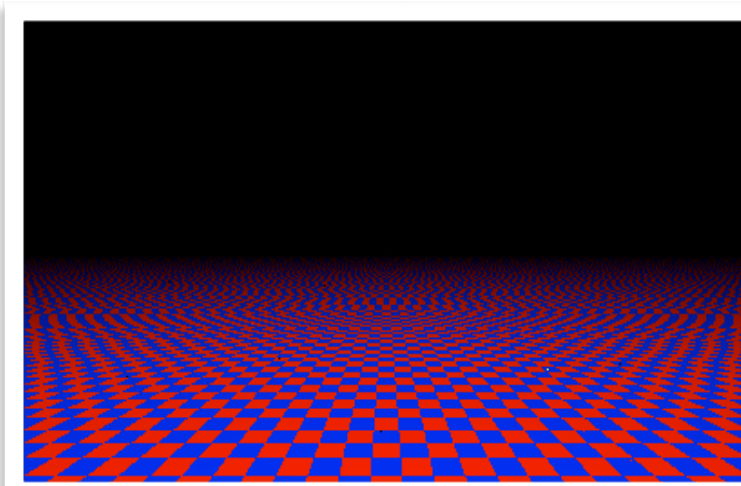
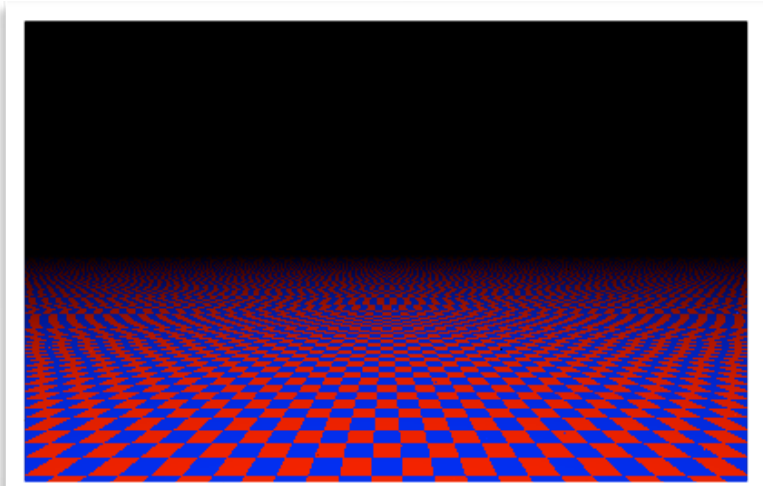
Strategies

- Approximate circuits
 - Adders, multipliers and other logical circuits can reduce hardware overhead, e.g. ignore the carry chain and thus allow all its sub-adders to perform addition operation in parallel.
- Approximate storage and memory
 - Truncating the lower-bits in floating point data
 - Less reliable memory, e.g. disabling any error detection and correction mechanisms
- Software-level approximation
 - Some iterations of loops can be skipped to achieve a result faster.
 - Task skipping, if run-time condition suggests that those tasks are not going to be useful
- Main challenge
 - Identification of the section of the application that can be approximated
 - Detailed domain knowledge needed

Point of View on Approximate Computing



Approximate Computing Example - Images



<https://www.cs.sjtu.edu.cn/~wuct/bd-it/slides/lec4.pdf>

Approximate Computing Example

